Name – KANAD DAS

Roll No. – B18733

University – RAMAKRISHNA MISSION VIVEKANANDA EDUCATIONAL & RESEARCH INSTITUTE

Program Course – BIG DATA ANALYTICS (BDA)

Program Release Date – 11 March 2019

Date of Submission – 01 April 2019

SUPERVISED LEARNING ALGORITHMS: TEXT CLASSIFICATION

Introduction

Aim: - This is an experiment on text classification using different supervised learning classifiers and their variants conducted on the Reuters-21578 dataset. The aim is to evaluate the best performance for each of the classifiers by properly tuning the parameters of each classifier so that the least error is recorded during the classification.

Procedure: - From the given training and test set for the dataset, we pre-process the data and perform text-cleaning methods for better classification. We remove the stop words and punctuations from both the training and test sets. We then tokenize each word using the count vectorizer function and perform a stemming algorithm to map each token to its root word. Next, we create a term frequency matrix by using Tfidf vectorizer for the classifiers to work on.

We train our model for a classifier using the training set and obtain an optimum set of parameters in order to achieve the best performance. We then use our model to classify the data points in the test set evaluate the performance of the model. Finally, we analyse and compare our results with the actual class labels of the test samples and come up with appropriate conclusions based on our comparisons.

Dataset: -

The original Reuters-21578 corpus originally contains 135 categories and the categories are overlapped, i.e., a document may exist in several categories. Hence, we consider the Mod Apte version of Reuters, which contains 12902 documents with 90 categories and the corpus is divided into training and test sets. For the given experiment, we are given the following 10 categories:

alum, barley, coffee, dmk, fuel, livestock, palm-oil, retail, soybean, veg-oil

Methods to be Used

For the data classification, we use the following classifiers:

1. **Multinomial Naive Bayes Classifier:** In text classification, we are given a description $d \in \mathbb{X}$ of a document, where $\mathbb{X}$ is the document space; and a fixed set of classes $\mathbb{C} = \{c_1, c_2, \ldots, c_J\}$. We are given a training set $\mathbb{D}$ of labelled documents (d, c), where $(d, c) \in \mathbb{X} \times \mathbb{C}$. The best class in NB classification is given as

$$c_{map} = \arg\max_{c \in C} P(c \mid d) = \arg\max_{c \in C} P(c) \prod_{1 \le k \le n_d} P(t_k \mid c)$$
$$= \arg\max_{c \in C} \left[\log P(c) + \sum_{1 \le k \le n_d} \log P(t_k \mid c) \right]$$

   where $(t_1, t_2, \ldots, t_{n_d})$ are the tokens in d that are part of the vocabulary we use for classification and n_d is the number of such tokens in d. The prior probabilities are estimated as

$$P(c) = \frac{N_c}{N}$$

   where N_c is the number of documents in class c and N is the total number of documents.

   The conditional probability $P(t \mid c)$ is estimated as the relative frequency of term t in documents belonging to class c:

$$P(t \mid c) = \frac{T_{ct}}{\sum_{t' \in V} T_{ct'}}$$

   where T_{ct} is the number of occurrences of t in training documents from class c and V is the set of vocabulary. [1]

2. **Bernoulli Naïve Bayes Classifier:** The naive Bayes classifier using this model estimates the conditional probability $P(t \mid c)$ as the fraction of documents of class c that contains the term t.
3. **K-nearest neighbour Classifier:** The k-nearest neighbour decision rule puts a test data point into a particular class, if the class has the maximum number of members among the k-nearest neighbours of the set of training samples, finding the nearest neighbours using a suitable distance function.
   For ensuring the best performance of the k-NN classifier, we need to choose a good k value and a suitable distance function. Choosing the optimal value for k is generally done by the method of c-fold cross validation.
   c-fold Cross Validation: The procedure has a single parameter called c that refers to the number of groups that a given data sample is to be split into. The procedure is as follows:
   a) Shuffle the dataset randomly.
   b) Split the dataset into c groups.
   c) For each unique group:
      i. Take the group as a hold out or test data set
      ii. Take the remaining groups as a training data set

iii. Fit a model on the training set and evaluate it on the test set

iv. Retain the evaluation score and discard the model

d) Summarize the skill of the model using the sample of model evaluation scores.

Choose that value of k for which the c-fold cross validation model has the highest score. As for the suitable distance function, there are several distance-measuring functions based on the type of variables given below:

a) Continuous variables:

$$\text{Euclidean Distance} = \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

$$\text{Manhattan Distance} = \sum_{i=1}^{k}|x_i - y_i|$$

b) Categorical variables:

$$\text{Cosine Distance} = 1 - \frac{\sum_{i=1}^{k} x_i y_i}{\sqrt{\sum_{i=1}^{k} x_i^2}\sqrt{\sum_{i=1}^{k} y_i^2}}$$

4. **Logistic Regression Classifier:** The logistic regression model represents the posterior probabilities of the *K* classes via linear functions in *x*, while at the same time ensuring that they sum to one and remain in [0, 1]. [2] For *K*=2(class 0 and class 1), we have to compute the posterior probabilities $P(y=0|x)$ and $P(y=1|x)$ and *x* is to be classified to the class, which has maximum posterior probability. We can estimate the posterior probabilities as follows:

$$P(y=1|x) = \frac{e^{(\beta_0+\beta_1 x)}}{1+e^{(\beta_0+\beta_1 x)}}, \forall x \qquad \cdots(\text{i})$$

$$P(y=0|x) = \frac{1}{1+e^{(\beta_0+\beta_1 x)}}, \forall x$$

Here $\vec{\beta} = [\beta_0, \beta_1]$ are the parameters and eqn (i) is known as the *sigmoid function* or *logistic function*. The $\vec{\beta}$ are estimated using the training samples by Newton's optimization method.

5. **Support Vector Machines:** The method finds a hyperplane to separate the data belonging to two different classes.[4]

If the data is linearly separable then the objective is to

$$\text{Minimize } \frac{1}{2}\vec{w}^T\vec{w}$$

$$\text{subject to } y_i\left(\vec{w}^T X_i + b\right) \geq 1, \quad \forall i = 1, 2, \ldots n$$

When the data is not linearly separable, the objective is to

$$\text{Minimize } \frac{1}{2}\vec{w}^T\vec{w} + C\sum_{i=1}^{n}\gamma_i$$

$$\text{subject to } \gamma_i \geq 0 \text{ and } y_i\left(\vec{w}^T X_i + b\right) \geq \left(1-\gamma_i\right),\ \forall i = 1, 2, \ldots n$$

Here γ_i is the proportional amount by which the prediction $\vec{x_i}$ is on the wrong side of the margin $\vec{w}^T X_i + b = 0$ and $C \in \left(0, \alpha\right]$. This is a constrained optimization problem. [3]

A test data point $\overrightarrow{X_0}$ can be classified by simply using the $Sign(\vec{w}^T X_0 + b)$. If $Sign(\vec{w}^T X_0 + b) < 0$ then $\overrightarrow{X_0}$ belongs to the negative class and if $Sign(\vec{w}^T X_0 + b) > 0$ then $\overrightarrow{X_0}$ belongs to the positive class.

Evaluation Criteria

There are different types of evaluation criteria to evaluate the performance of a classifier using the ground truths of the given data set. Let us assume there are two classes in the data set, say, positive and negative. A classifier makes error when a positive sample is predicted as negative and a negative sample is predicted as positive. The same may be observed from the table below:

		Predicted Class	
		Positive	Negative
Actual Class	Positive	TP	FN
	Negative	FP	TN

where,
TP (*true positive*) = number of data points correctly predicted to the positive class.
FP (*false positive*) = number of data points that originally belong to the negative class, but predicted as positive (i.e., *falsely predicted as positive*).
FN (*false negative*) = number of data points that originally belong to the positive class, but predicted as negative (i.e., *falsely predicted as negative*).
TN (*true negative*) = number of data points correctly predicted to the negative class.
The above contingency table is known as the *confusion matrix*.
To measure the effectiveness of a classifier, we compute the following measures:

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

Again, there are two conventional methods to evaluate the performance of a classifier aggregated over all classes, namely *macro-averaging* and *micro-averaging*.
The macro averaged precision and recall for a set of m classes are computed as follow:

$$\text{Macro-averaged Precision} = \frac{1}{m}\sum_{i=1}^{m}\frac{TP_i}{TP_i + FP_i}$$

$$\text{Macro-averaged Recall} = \frac{1}{m}\sum_{i=1}^{m}\frac{TP_i}{TP_i + FN_i}$$

Here TP_i stands for true positive of the i^{th} class, FP_i stands for the false positive of the i^{th} class, FN_i stands for the false negative of the i^{th} class and TN_i stands for the true negative of the i^{th} class.
The micro averaged precision and recall for a set of m classes are computed as follow:

$$\text{Micro-averaged Precision} = \frac{\sum_{i=1}^{m} TP_i}{\sum_{i=1}^{m} TP_i + \sum_{i=1}^{m} FP_i}$$

$$\text{Micro-averaged Recall} = \frac{\sum_{i=1}^{m} TP_i}{\sum_{i=1}^{m} TP_i + \sum_{i=1}^{m} FN_i}$$

The f-measure combines recall and precision with an equal weight in the following form:

$$F - measure = \frac{2 \times precision \times recall}{precision + recall}$$

The closer the values of precision and recall, the higher is the f-measure. A high f-measure value is desirable for good classification.

Analysis and Results

The results obtained from running the different types of classifiers using the pipeline and grid search (with cross-validation) tools for the dataset are as follows.

1. **Bernoulli Naïve Bayes Classifier:** The optimum set pf parameters of the classifier is given as

   ```
   {'clf__alpha':0.01,'vect__max_df':0.52,'vect__ngram_range':(1, 2)}
   ```

   The confusion matrix is given by

$$CM = \begin{bmatrix} 12 & 2 & 5 & 2 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 1 & 25 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 2 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 5 & 2 & 0 & 0 & 0 & 1 \\ 0 & 2 & 2 & 0 & 0 & 16 & 0 & 0 & 1 & 3 \\ 0 & 0 & 2 & 0 & 0 & 0 & 4 & 0 & 1 & 3 \\ 0 & 0 & 2 & 0 & 0 & 0 & 4 & 0 & 1 & 3 \\ 0 & 9 & 0 & 1 & 0 & 1 & 1 & 0 & 12 & 9 \\ 0 & 5 & 3 & 0 & 0 & 0 & 5 & 0 & 7 & 17 \end{bmatrix}$$

2. **Multinomial Naïve Bayes Classifier:** The optimum set of parameters of the classifier is given as

   ```
   {'clf__alpha':0.01,'vect__max_df':0.2,'vect__ngram_range':(1, 2)}
   ```

   The confusion matrix is given by

$$CM = \begin{bmatrix} 15 & 0 & 3 & 0 & 0 & 0 & 1 & 0 & 0 & 4 \\ 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 1 \\ 0 & 0 & 28 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 3 & 0 & 5 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 17 & 0 & 0 & 2 & 4 \\ 0 & 0 & 1 & 0 & 0 & 0 & 3 & 0 & 0 & 6 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 25 & 5 \\ 0 & 0 & 2 & 0 & 0 & 0 & 3 & 0 & 12 & 20 \end{bmatrix}$$

3. **KNN Classifier:** The optimum set of parameters of the classifier is given as

   ```
   {'clf__metric':'cosine','clf__n_neighbors':27,'vect__max_df':0.48
   ,'vect__ngram_range':(1, 1)}
   ```

   The confusion matrix is given by

$$CM = \begin{bmatrix} 15 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 3 & 3 \\ 0 & 9 & 1 & 0 & 0 & 1 & 0 & 0 & 3 & 0 \\ 0 & 0 & 28 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 3 & 0 & 0 & 2 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 16 & 0 & 0 & 1 & 5 \\ 0 & 0 & 1 & 0 & 0 & 0 & 3 & 0 & 1 & 5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 1 & 1 & 0 & 24 & 4 \\ 0 & 0 & 4 & 0 & 0 & 0 & 3 & 0 & 11 & 19 \end{bmatrix}$$

4. **Logistic Regression Classifier:** The optimum set of parameters of the classifier is given as

```
{'clf__C':215.443469003188,'vect__max_df':0.4,'vect__ngram_range:(1,1)}
```

The confusion matrix is given as

$$CM = \begin{bmatrix} 19 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 3 \\ 0 & 11 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 \\ 0 & 0 & 27 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 6 & 2 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 20 & 0 & 0 & 1 & 3 \\ 0 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 24 & 6 \\ 0 & 0 & 2 & 0 & 0 & 2 & 2 & 0 & 8 & 23 \end{bmatrix}$$

5. **Support Vector Machine:**
   a. **Without using kernel:** The optimum set of parameters for the classifier is given as

```
{'clf__C':46.41588833612782,'clf__tol':0.0001,'vect__max_df':0.4,
'vect__ngram_range':(1, 2)}
```

The confusion matrix is given as

$$CM = \begin{bmatrix} 21 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 11 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 \\ 0 & 0 & 28 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 7 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 20 & 0 & 0 & 1 & 3 \\ 0 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 24 & 6 \\ 0 & 1 & 2 & 0 & 0 & 1 & 2 & 0 & 9 & 22 \end{bmatrix}$$

b. **<u>Using linear kernel:</u>** The optimum set of parameters for the classifier is given as

```
{'clf__C':2.154434690031882,'clf__tol': 1.0,'vect__max_df':0.8,'
vect__ngram_range':(1, 2)}
```

The confusion matrix is given as

$$CM = \begin{bmatrix} 16 & 0 & 1 & 0 & 0 & 1 & 0 & 2 & 0 & 3 \\ 0 & 9 & 0 & 0 & 0 & 1 & 0 & 0 & 3 & 1 \\ 0 & 0 & 28 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 7 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 20 & 0 & 0 & 1 & 3 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 9 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 23 & 7 \\ 0 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 6 & 28 \end{bmatrix}$$

b. **<u>Using non-linear kernel:</u>** The optimum set of parameters for the classifier is given as

```
{'clf__C':3.593813663804626,'clf__kernel':'sigmoid','clf__tol':
1.0,'vect__max_df':0.48,'vect__ngram_range':(1, 2)}
```

The confusion matrix is given as

$$CM = \begin{bmatrix} 17 & 0 & 0 & 0 & 0 & 2 & 0 & 2 & 0 & 2 \\ 0 & 10 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 1 \\ 0 & 0 & 28 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 7 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 19 & 0 & 0 & 1 & 3 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 9 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 3 & 0 & 0 & 23 & 0 \\ 0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 8 & 25 \end{bmatrix}$$

The summary of the evaluation criteria for the different classifiers is presented in the following table:

Table 1: Comparative study of the micro-averaged and macro-averaged measures for the various classifiers

Evaluation Criteria		**Classifiers**						
		Bernoulli Naïve Bayes Classifier	**Multinomial Naïve Bayes Classifier**	**KNN Classifier (k=27)**	**Logistic Regression Classifier**	**Support Vector Machine**		
						without kernel	**with linear kernel**	**with non-linear kernel**
Micro-averaged	Precision	0.56	0.66	0.65	0.74	0.76	0.74	0.72
	Recall	0.56	0.66	0.65	0.74	0.76	0.74	0.72
	F-measure	0.56	0.66	0.65	0.74	0.76	0.74	0.72
Macro-averaged	Precision	0.63	0.70	0.74	0.82	0.78	0.73	0.72
	Recall	0.57	0.56	0.61	0.71	0.75	0.71	0.71
	F-measure	0.57	0.60	0.61	0.74	0.74	0.70	0.69

Analysis: From the above table, we can see that the micro-averaged measures of the **Bernoulli Naïve Bayes Classifier** (= **0.56**) are the least among the classifiers. The optimum classifier out of the five classifiers is **Support Vector Machine (without using kernel)**, having the micro-averaged measures equal to **0.76**. We are considering the micro-averaged measures for comparison since we have a class imbalance in the dataset. Hence, we are prioritizing the classes with higher number of samples as compared to those having a smaller number of samples.

If we consider the macro-averaged f-measures, we find that the **Bernoulli Naïve Bayes Classifier** has the least value (=**0.57**) among the classifiers. Also, the optimum classifiers are the **Logistic Regression Classifier** and the **Support Vector Machine (without using kernel)**, both having f-measure = **0.74**. The macro-averaged measures give equal weightage to the classes and thus give a more effective measures on the smaller classes.

Conclusion

From the above observations, we see that the Bernoulli Naïve Bayes Classifier perform very poorly than the other classifiers. This seems justified as the Bernoulli NB classifier considers only the presence or absence of a term in the document, rather than considering the number of occurrences and hence cannot effectively classify long datasets. Compared to this, the Multinomial Naïve Bayes Classifier considers the number of occurrences of a term in the document and hence gives a decent performance for this long dataset.

The k-nearest neighbours classifier performs at par with the Multinomial NB classifier, even though it is generally not a suitable algorithm for text classification. Choosing an optimal value of *k* to improve the performance of the classifier is still a disadvantage of this algorithm. As a solution to this problem, one can choose different values of *k* for different classes in the document, according to their distribution in the training set. [5]

The logistic regression classifier is the second-best classifier and rightly so since it is one of the most popular classifiers in text classification. It contains only one hyperparameter C which is the inverse of regularization strength (λ) and serves as a penalty parameter to the cost function. For the given dataset, the parameter C has a high value (=**215.443**) and thus the model tends to overfit the data. This can be avoided by training the classifier on more data, ensuring that the training data is clean and relevant. Another solution to overfitting suggests the reduction in the number of features, either by manually removing irrelevant features or by using the in-built feature selection in the classifier.

The support vector machine proves to be the most efficient classifier in terms of the micro-averaged measures, and it implements two types of classification techniques: (i) simple SVM (LinearSVC) assuming the data to be linearly separable and (ii) kernel-based techniques which implements both linear and non-linear kernels. From Table 1, we see that the simple SVM has the highest micro-averaged measures with C = **46.415**, thus eliminating the chance of overfitting as well as underfitting. Compared to this, the kernel-based linear SVM performs less effectively with f1-score (micro) = **0.74** and C = **2.154**. The small value of C indicates a tendency of underfitting by the model. To avoid this, more features can be included, or we can increase the flexibility of the model by introducing few parameters. As for the non-linear kernel-based techniques, the model performs satisfactorily overall but with a low value of C (=**3.594**), it too suffers from underfitting. Another significant point to be noted is that a comparison between the linear and no-linear models of the Support Vector Machine shows that the dataset is linearly separable.

References

[1] C. D. Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval*. Cambridge University Press, New York, 2009.

[2] Jerome H. Friedman, Robert Tibshirani, and Trevor Hastie. *The Elements of Statistical Learning.* Springer, second edition, 2008.

[3] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[4] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern Recognition*. Elsevier, fourth edition, 2008.

[5] Li, Baoli & Yu, Shiwen & Lu, Qin. (2003). An Improved k-Nearest Neighbor Algorithm for Text Categorization.